# A natural approach to sample selection in binary classification

Jesús Cid-Sueiro, *Member, IEEE*, Alicia Guerrero-Curieses

*Abstract*—

**This paper considers the problem of training neural networks to solve general decision problems. We present a family of learning algorithm based on the minimization of a particular family of *Strict Sense Bayesian* (SSB) cost functions, which are those that provide estimates of the posterior probabilities of the classes. This family of cost functions is selected in order to provide the most accurate estimates for the probability values near the decision thresholds. It is shown that learning algorithms based on this approach employ a sample selection strategy: samples near the decision boundary are the most relevant during learning**

## I. Introduction

Consider a general soft-decision-based classifiers (SDC), which makes decisions in two steps: (1) it computes a *soft* decision (SD) (usually, a differentiable function of the classifier parameters), and (2) it uses the SD to make a *hard* decision (HD), which is an element of a finite set of classes.

Classical decision theory shows that the HD are functions of the posterior class probabilities. This suggests that the SD system should compute or estimate them in order to achieve the best performance. The problem of estimating posterior class probabilities by learning from examples has received some recent interest in the neural network literature. General conditions for cost functions that minimize to conditional class probabilities are given in [1] and [2], mainly for single-output classifiers. In [3] and [4], the analysis is extended to multi-output classifiers. This approach splits the design of the SDC in two independent steps: learning to estimate posterior probabilities and establishing a decision criteria. An additional advantage of estimating posterior probabilities is that they provide a confidence measure of the classifier decisions.

However, it is well known that, in order to make optimal decisions it is not required to estimate class probabilities. Essentially, only the optimal decision boundary must be estimated. Moreover, in practice the classifier cannot map exactly the true posterior probabilities, and using the finite resources of the classifier to estimate posterior probabilities far from the decision boundary may affect the decision performance. In summary, the classifier performance is improved when decision and learning are taken as a whole.

Jesús Cid-Sueiro and Alicia Guerrero-Curieses are with the Dpto. de Tecnologías de las Comunicaciones, EPS, Universidad Carlos III de Madrid, Avda. de la Universidad, 30, 28919 Leganés-Madrid, Spain.

Most classification algorithms pursue the goal of minimizing the expected number of errors. However, decision theory shows that this is not necessarily an adequate decision criteria. In general, wherever there is an action associated to every possible classifier decision, different kinds of errors entail different risks. For instance, the consequences of a false alarm in a radar detection system can be dramatically different than that of a detection error. Similar comments apply to medical diagnosis or financial data analysis systems.

To take into account the cost of each kind of error, minimizing other risk measure than the error probability may be preferred. The design of structures and algorithms for non-MAP decision problems is the focus of this paper. Our approach is based on the analysis of *Strict Sense Bayesian Cost Functions* [4] which are those that reach its minimum when the soft decisions are equal to the posterior probabilities of the classes. We show that SSB cost functions that provide accurate estimates at given values of the posterior probabilities are useful to solve general decision problems. Moreover, we show that learning algorithms based on the minimization of such cost functions behave like sample selectors that focus the learning process in the samples near decision boundaries.

## II. Learning and Decision

Consider the sample set $\mathcal{S} = \{(\mathbf{x}_k, d_k), k = 1, \ldots, K\}$, where $\mathbf{x}_k$ is an element of observation space $\mathcal{X}$ (typically, a measurable subset of $\mathbb{R}^N$, and $d_k$ is one of two possible classes $d_k \in \{0, 1\}$. The elements in $\mathcal{S}$ have been generated independently according to probability model $p(\mathbf{x}, d)$, where $\mathbf{x} \in \mathcal{X}$ and $d \in \{0, 1\}$

Consider a non-linear classifier whose *soft decision* is given by $y = \mathbf{f}_\mathbf{w}(\mathbf{x})$, where $\mathbf{f}_\mathbf{w} : \mathcal{X} \to \mathcal{P}$ is a function with parameters $\mathbf{w}$ and $\mathcal{P} = [0, 1]$. Since soft decisions are elements of $\mathcal{P}$, they can be interpreted as probabilities. Any single output neural network with a sigmoid nonlinearity as a final step is an example of this kind of classifiers.

The *hard* output of the classifier is allowed to perform any non-linear function $h : \mathcal{P} \to \{0, 1\}$ transforming probabilities into decisions. In the particular case of MAP decision problems, $h$ computes the integer (0 or 1) that is closest to $y$.

Parameters $\mathbf{w}$ must be learned using the samples in $\mathcal{S}$, with the goal of optimizing some decision criteria. We assume that learning is based on the minimization of some

empirical estimate of

$$E\{C(y,d)\} = \int C(y,d)dP(\mathbf{x}) \tag{1}$$

where $C(y,d)$ is a *cost function*. The design of cost functions for specific decision problems is the focus of this paper.

### A. Decision theory

According to Van Trees [5], in any classification problem we can associate a cost $c_{ij}$ of deciding in favor of class $i$ when the true class is $j$. For a binary case, the risk associated to decision $i$ (0 or 1) is defined as

$$R_i = c_{i0}P(d=0 \mid \widehat{d}_i = 1) + c_{i1}P(d=1 \mid \widehat{d}_i = 1) \tag{2}$$

The overall risk is defined as,

$$R = P(\widehat{d}=0)R_0 + P(\widehat{d}=1)R_1 \tag{3}$$

Decision rules should be designed in order to minimize the overall risk. It is well-known that such a minima is reached for a decision machine computing, for every sample $\mathbf{x}$, class $c$ given by

$$c = \arg\min_i \{c_{i0}p_0 + c_{i1}p_1\} \tag{4}$$

where $p_i = P(i \mid \mathbf{x})$. Thus, the decision boundary separating classes $k$ and $l$ is given by the equation

$$c_{00}p_0 + c_{01}p_1 = c_{10}p_0 + c_{11}p_1 \tag{5}$$

which is equivalent to

$$p_1 = \frac{c_{00} - c_{10}}{c_{00} - c_{10} + c_{11} - c_{01}} \tag{6}$$

In particular, taking $c_{ii} = 0$ and $c_{01} = c_{10}$ risk (3) is equal to the error probability and the decision rule becomes

$$c = \arg\max_i \{p_i\} \tag{7}$$

which is the decision rule of the *Maximum A Posteriori* (MAP) classifier.

### B. Learning

The learning problem consists of estimating parameters $\mathbf{w}$ of the SDC such that decisions are optimal. The SD system should compute some sufficient statistics for the HD system. Since the decision rules analyzed in the previous section depend on the posterior class probabilities, in order to get optimal performance it is sufficient to get $y = P(d=1 \mid \mathbf{x})$. We will say that a cost function being minima when soft decisions are class probabilities is *Wide Sense Bayesian*. Since this is not a necessary condition to get optimal decision, we will allow the cost function to reach the same minima for other output values. A formal definition is given below.

*Definition 1:* Wide Sense Bayesian cost function

Cost function $C : \mathcal{P} \times \{0,1\} \to \mathbb{R}$, is said to be *Wide Sense Bayesian*, or WSB, if $E\{C(y,d) \mid \mathbf{x}\}$ has a global minimum when $y$ is the posterior class probability for every $\mathbf{x} \in \mathcal{X}$, i.e.,

$$y = P(d=1 \mid \mathbf{x}) \tag{8}$$

It is interesting to distinguish, among all WSB cost functions, those providing a unique minimun at posterior probability values. They are defined in the following

*Definition 2:* Strict Sense Bayesian cost function [4]

Cost function $C : \mathcal{P} \times \{0,1\} \to \mathbb{R}$, is said to be *Strict Sense Bayesian*, or SSB, if it is WSB and $E\{C(y,d) \mid \mathbf{x}\}$ has a unique minimum.

Note that, the only constraint imposed to $C$ in the previous definitions is that of being a function of $y$ and $d$, that is, the entire influence of the network parameters on the cost function is given by $y$. Note, also, that the definition is independent of the classifier structure, and does not deal with the possible limitations of a given classifier to compute the true probability map. Anyway, if the classifier has not enough power to compute the posteriori probability map, we want to find parameters providing the *best* approximation, according to some criterion.

Now, we provide a complete characterization of WSB and SSB cost functions

*Theorem 1:* Cost function $C : \mathcal{P} \times \{0,1\} \to \mathbb{R}$ is WSB iff it can be expressed in the form

$$C(y,d) = \int_d^y g(z)(z-d)dz + r(d) \tag{9}$$

where $g : \mathcal{P} \to \mathbb{R}$ is a nonnegative function and $r : \{0,1\} \to \mathbb{R}$ is an arbitrary function.

The proof follows the same steps than that of Th. 1 in [4], and is omitted here. Note that $r$ is irrelevant for optimization purposes, and we can take $r = 0$. In this case, the cost function is said to be *canonical*.

*Theorem 2:* A cost function $C : \mathcal{P} \times \{0,1\} \to \mathbb{R}$ is SSB iff it is WSB and $g(y)$ is strictly positive in $\mathcal{P}$.

There are two well-known examples of SSB cost functions: If $g(y) = 1/(y(1-y))$,

$$C(y,d) = -d\log y - (1-d)\log(1-y) \tag{10}$$

which is the *cross entropy* (or *relative entropy*) cost function. If $g(y) = 2$, we obtain

$$C(y,d) = (d-y)^2 \tag{11}$$

which is the square error.

### C. Divergences

Although all SSB cost functions have the same minimum at $y = p$ for every input $\mathbf{x}$, the selection of a particular cost function is of major importance. In particular, if the SDC is not complex enough to compute the true data probability map for any parameter values, different SSB cost functions provide different posterior probability estimates. It can be shown that, for every SSB cost

function $C$, a divergence measure $D : \mathcal{P}^2 \to \mathbb{R}$ between probability distributions $p$ and $y$ can be defined as

$$D(p,y) = E\{C(y,d) - C(p,d) \mid \mathbf{x}\} = \int_p^y g(z)(z-p)dz \tag{12}$$

In the following, $D$ will be called the SSB divergence derived from $C$. Since $C(p,d)$ is independent of the classifier parameters, minimizing the mean cost is equivalent to minimizing its derived divergence. Therefore, we can say that any SSB cost function estimates the posterior probabilities minimizing a sample average estimate of the *mean SSB divergence*, $E\{D(p,y)\}$. In order to select a particular cost function, the knowledge of its derived divergence becomes relevant.

## III. Sensitivity of a SSB divergence

In general, posterior probability $p$ is an unknown function of the observation $\mathbf{x}$. Since our final goal is to minimize an overall risk function as defined in (3), we should maximize the accuracy of the probability estimates near the decision threshold given by (6).

The SSB divergence should have maximum sensitivity around probability vectors near the decision boundary. The sensitivity is defined here as the directional second derivative of the divergence at given point.

*Definition 3:* The *sensitivity* of a SSB divergence at $p = p_0 \in \mathcal{P}$ is

$$s(p_0) = \left.\frac{d^2 D(p_0,y)}{dy^2}\right|_{y=p_0} \tag{13}$$

The sensitivity measures the velocity of change of the divergence around $\mathbf{p}_0$. The sensitivity is always positive, since $D(y,p)$ is a convex function of $y$ at $y = p$, for any $p \in \mathcal{P}$. Moreover, it is easy to show that

$$s(p_0) = g(p_0) \tag{14}$$

Therefore, our goal is finding a function $g$ such that the sensitivity is maximized at a given boundary. It is illustrative to see that conventional cost functions are not useful for this purpose. In particular, for the square error, $s(p_0)$ is constante and, for the cross entropy,

$$s(p_0) = \frac{1}{p_0(1-p_0)} \tag{15}$$

which has no maximum in $(0,1)$.

## IV. Designing SSB cost functions

Consider the family of SSB cost functions $C_{q,r}(y,d)$ given by

$$g_{q,r}(y) = a\left(y^q(1-y)^{(1-q)}\right)^r \tag{16}$$

where $r$ is some constant parameter, and $a = \beta(1+qr, 1+(1-q)r)^{-1}$ (where $\beta$ is the beta functioin) is a normalizing constant ensuring that $g_{q,r}$ has unity area. Constant $a$ is irrelevant for learning, but it is included for the convenience of the theoretical analysis that follows. For any $r > 0$ $C_{q,r}$ has maximal sensitivity at $y = q$. Thus, selecting different values of $q$ we can train a neural network to solve different decision problems. Parameter $r$ controls the sharpness of $g$ around $q$.

There are two particular cases of interest: $C_{0.5,0}$ is the square error and $C_{0.5,-1}$ is the cross entropy. However, since $r$ is not positive, these cost functions seem to be not adequate for optimizing decisions.

Note that $C_{q,r}$ does not need to be computed for gradient-based learning algorithms. For instance, the stochastic gradient learning rule for $g_{q,r}$ and a general neural net with parameters $\mathbf{w}$ is given by

$$\mathbf{w}(k+1) = \mathbf{w}(k) - (d-y)g_{q,r}(y)\nabla_{\mathbf{w}} y \tag{17}$$

In this rule $g_{q,r}$ behaves like a sample selector: for high $r$ only samples providing outputs near $q$ change signifcantly the weight vector. The sample selection capability of some WSB learning rules is more evident for the alternative family

$$g(y) = I_{|y-q|\le\epsilon} \tag{18}$$

where $\epsilon < 1$ is some constant parameters, and $I$ is an indicator function, which is equal to 1 if the condition is verified, and 0 otherwise. In this case, stochastic gradient learning rules become

$$\mathbf{w}(k+1) = \mathbf{w}(k) - (d-y)I_{|y-q|\le\epsilon}\nabla_{\mathbf{w}} y \tag{19}$$

which shows that only samples in the neighborhood of the decision boundary are used during learning.

### A. Cost functions to optimize decisions

The family $C_{q,r}$ is assymptotically optimal as $r$ goes to infinity. This is shown by the following:

*Theorem 3:* Let $C_{q,r}(y,d)$ the SSB cost function given by $g_{q,r}(y)$. Then,

$$C_q(y,d) = \lim_{r\to\infty} C_{q,r}(y,d) = \begin{cases} 0, & \text{if } d = \widehat{d} \\ q, & \text{if } 0 = d \neq \widehat{d} \\ 1-q, & \text{if } \widehat{d} \neq d = 1 \end{cases} \tag{20}$$

where $\widehat{d}$ is a hard decision for threshold $q$

$$\widehat{d} = \begin{cases} 0, & \text{if } y < q \\ 1, & \text{if } y > q \end{cases} \tag{21}$$

That is, in the limiting case, $E\{C_{q,r}\}$ becomes equivalent to the risk function $R$ given by $c_{00} = c_{11} = 0$, $c_{10} = q$ and $c_{01} = 1-q$. (Additionally, it can be shown that $C_q$ is the WSB (but not SSB) cost function defined by

$$g(y) = \delta(y-q) \tag{22}$$

where $\delta$ is the Dirac delta function).

Although $C_{q,r}$ is assymptoticaly optimal, it can be shown that, for high $r$, the error hypersurfaces tend to have highly flat regions. Therefore, local minima problems are frequent. This suggest the following learning

strategy: start learning with a low $r$ (for instance, $r = -1$ (cross entropy) which is known to be well-behaved, and increase $r$ as learning proceeds. In the following section, some simulations will serve to analyze the performance of this method and, also, to compare it with other approaches to the problem of training learning machines to solve general decision problems [6].

## V. Experiments

The experiments were performed using two different real data sets: the ionosphere database of the UCI Machine Learning Repository and data extracted from a Landsat image of a 47.2 $\times$ 44 $Km^2$ area in the North-West of Spain (the same used in [7]).

The classifier was the softmax perceptron with soft decision, given by

$$y_{ij} = \frac{\exp(\mathbf{w}_{ij}^T \mathbf{x})}{\sum_l \sum_m \exp(\mathbf{w}_{lm}^T \mathbf{x})} \tag{23}$$

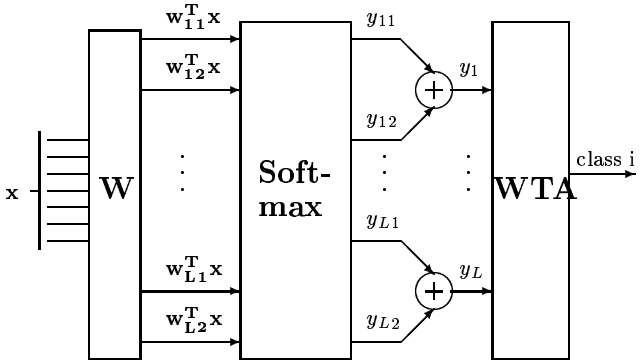


Fig. 1. Neural network architecture to estimate the posterior class probabilities.

Fig. 1 represents the neural network architecture used to classify. $\mathbf{x}$ is a sample, and $\mathbf{w_{ij}}$ is the weight vector of perceptron $j$ of class $i$ ($i = 1, 2$ and $j = 1, 2$ in these experiments). Stochastic gradient learning rules were applied: initial adaptation step $\mu_0 = 1$ decreasing according to $\mu_k = \mu_0/(1 + k/k_0)$, where $k$ is the iteration number and $k_0 = 1000$. To avoid some cases of convergence to wrong local minima, we used a simple method: for each simulation, each training process was performed 5 times, and the case with the lowest value of the corresponding risk funcional on the training set was selected to measure errors.

The ionosphere database is made up of 351 instances. The first 200 instances have been used for training, and the other 151 for testing. Each instance has 34 attributes and one target ("good",if radar return showes evidence of same type of structure in the ionosphere) or "bad", if not).

Error rates in the ionosphere test set for each value of $r$ (-1 (cross entropy), 0, 1 and 2), used in consecutive training, are shown in Table I. In order to get good results only one perceptron per class has been used. With a second training, the error probability has been reduced in a 65.09%. Posterior training with higher values of $r$ did not improve the classification in this case.

| $r$=-1 | $r$=0 | $r$=1 | $r$=2 | $r$=3 |
|---|---|---|---|---|
| 10.6 | 3.7 | 3.7 | 3.7 | 3.7 |

TABLE I
Percentage of errors in the ionosphere test set for different values of $r$

In the case of the Landsat Image, the objetive is to classify two different types of terrain, *eucalyptus* and *pine*, present in the image. A database of 2211 was split in 1710 training samples and 501 test samples.

Fig. 2 show the error rates in the image test set for the same values of $r$ of the previous example and using two perceptrons per class. In this case, five consecutive trainings were necessary to get an improvement of 8.52% in the final classification. It is less significant than in the previous example because the training set has four times more samples of eucalyptus than of pine. Moreover, when we want to classify more than two types of terrains (multiclass classification), pine and eucalyptus are two of the classes where high number of mistakes are committed.

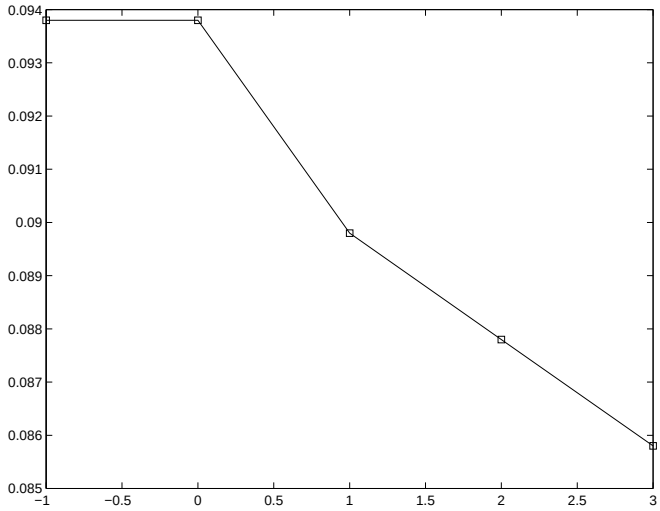


Fig. 2. Error probability in the image test set vs. value of $r$ used for training

## VI. Conclusions

This paper analyzes the structure of SSB cost functions (those that provide estimates of the *a posteriori* probabilities of the classes), showing that SSB cost functions that provide accurate estimates at given values of the posterior probabilities are useful to solve general decision problems. We have provided a general formula for a family of SSB cost functions that has maximal sensitivity at binary decision boundary.

The previous simulation results show that learning algorithms based on the minimization of such cost functions behave like a sample selector, that focus the learn-

ing process in the samples near the decision boundary. The structure and behavior of SSB cost families that have maximum sensitivity at multiclass decision boundaries are being studied by the authors.

## References


[1] B. Pearlmutter and J. Hampshire, "Equivalence proofs for multi-layer perceptron classifiers and the bayesian discriminant function," in *Proc. of the 1990 Connectionist Models Summer Shool*, San Diego, CA, 1990, Morgan Kauffmann.

[2] J.W. Miller, R. Goodman, and P. Smyth, "On loss functions which minimize to conditional expected values and posterior probabilities," *IEEE Transactions on Information Theory*, vol. 39, no. 4, pp. 1404–1408, July 1993.

[3] M. Saerens, "Non-mean square error criteria for the training of learning machines," in *Proc. of 13th Intl. Conf. on Machine Learning (ICML'96)*, Bari, Italy, 1996, pp. 427–434.

[4] J. Cid-Sueiro, J.I. Arribas, S. Urbán Mu noz, and A.R. Figueiras-Vidal, "Cost functions to estimate a posteriori probabilities in multi-class problems," *IEEE Transactions on Neural Networks*, vol. 10, no. 3, pp. 645–656, May 1999.

[5] H.L. Van Trees, *Detection, Estimation and Modulation Theory*, vol. I, John Wiley, New York, 1968.

[6] L. Tarassenko, *A Guide to Neural Computing Applications*, Arnold, London, 1998.

[7] J.L. Alba, L. Docío, D. Docampo, and O.W. Márquez,, "Growing Gaussian mixtures network for classification applications," *Signal Processing*, vol. 76, no. 1, pp. 43–60, July 1999.